Karnataka PGCET MCA 2025 Question Paper with Solutions

Time Allowed :2 Hours 30 Minutes	**Maximum Marks :**100	**Total Questions :**75

General Instructions

Read the following instructions very carefully and strictly follow them:

1. The duration of the test is 150 minutes
2. MCA PGCET Paper is divided into 2 parts- Part-A and Part-B
3. Each question will be asked in a multiple-choice (Objective) type format, wherein each question will have four different options
4. In Part A, there will be 50 questions that will carry one mark each
5. In Part B, there will be 25 questions that will carry two marks each. Hence, there will be 75 questions and the total mark is 100

1. Which of the following data structures is most suitable for implementing a priority queue?

(A) Array

(B) Stack

(C) Binary Heap

(D) Linked List

Correct Answer: (C) Binary Heap

Solution:
A priority queue is an abstract data type that stores elements with associated priorities.

Elements with higher priority are dequeued before elements with lower priority.

Let's analyze the time complexities of priority queue operations for each data structure.

For an array-based implementation, insertion can be $O(1)$, but finding the element with the highest priority would take $O(n)$ time.

For a stack, the order is Last-In-First-Out (LIFO), which is not based on priority.

For a linked list, finding the correct position to insert based on priority takes $O(n)$ time.

A Binary Heap is a specialized tree-based data structure that satisfies the heap property.

In a Binary Heap, both insertion (enqueue) and removal of the highest-priority element (dequeue) can be performed in $O(\log n)$ time.

This logarithmic time complexity is significantly more efficient than the linear time complexity of array or linked list implementations.

Therefore, a Binary Heap is the most suitable data structure for implementing a priority queue.

> Quick Tip
>
> For priority queue implementations, always associate Binary Heap with $O(\log n)$ efficiency for insertion and deletion, making it superior to array or linked list-based implementations which are typically $O(n)$.

---

2. What is the time complexity of accessing an element in a hash table (assuming a good hash function)?

(A) $O(1)$

(B) $O(\log n)$

(C) $O(n)$

(D) $O(n^2)$

Correct Answer: (A) $O(1)$

Solution:
A hash table is a data structure that maps keys to values.

It uses a hash function to compute an index into an array of buckets or slots.

To access an element, the hash function is applied to the key to directly calculate the index where the value is stored.

The phrase "assuming a good hash function" implies that the function distributes keys uniformly across the buckets, minimizing collisions.

Since the index is computed directly and array access is a constant-time operation, the overall time complexity is constant.

Therefore, the average time complexity for accessing an element in a hash table is $O(1)$.

> Quick Tip
>
> Remember that the $O(1)$ complexity for hash tables is the average case. In the worst case, where all keys hash to the same index (many collisions), the time complexity can degrade to $O(n)$. The phrase "good hash function" signals that you should consider the average case.

---

3. Which of the following is a correct statement regarding object-oriented programming (OOP)?

(A) OOP is based on functions rather than data

(B) In OOP, objects can have both data and methods

(C) In OOP, the program is written as a sequence of instructions

(D) OOP does not support inheritance

Correct Answer: (B) In OOP, objects can have both data and methods

Solution:
Let's evaluate each statement about Object-Oriented Programming (OOP).

(A) This statement is incorrect. It describes procedural programming. OOP is centered around objects, which encapsulate both data and behavior.

(C) This statement is incorrect. It also describes procedural or imperative programming, where programs are a sequence of commands. OOP programs are structured as a collection of interacting objects.

(D) This statement is incorrect. Inheritance is one of the fundamental principles of OOP, allowing a new class to inherit properties and methods from an existing class.

(B) This statement is correct. The core concept of OOP is the object, which is an instance of a class. An object bundles data (attributes) and the functions (methods) that operate on that data.

Quick Tip

The four pillars of OOP are Encapsulation, Abstraction, Inheritance, and Polymorphism. Any statement that contradicts one of these core principles is incorrect. Encapsulation is the bundling of data (attributes) and methods (functions) into a single unit, an object.

---

4. Which of the following sorting algorithms has the worst-case time complexity of $O(n^2)$?

(A) Merge Sort

(B) Quick Sort

(C) Bubble Sort

(D) Heap Sort

Correct Answer: (C) Bubble Sort

Solution:
Let's analyze the worst-case time complexity for each sorting algorithm listed.

(A) Merge Sort: This algorithm consistently divides the array and merges, resulting in a worst-case time complexity of $O(n \log n)$.

(B) Quick Sort: While its average-case complexity is $O(n \log n)$, its worst-case complexity is $O(n^2)$, which occurs with poor pivot selection (e.g., on an already sorted array).

(D) Heap Sort: This algorithm uses a heap data structure and has a consistent worst-case time complexity of $O(n \log n)$.

(C) Bubble Sort: This algorithm repeatedly compares adjacent elements. In the worst-case scenario (a reverse-sorted array), it performs nested loops, leading to a time complexity of $O(n^2)$. Its average case is also $O(n^2)$.

Both Quick Sort and Bubble Sort have a worst-case of $O(n^2)$. However, Bubble Sort is fundamentally an $O(n^2)$ algorithm for both average and worst cases, whereas the worst case for Quick Sort is an exception. Thus, Bubble Sort is the most definitive answer representing this complexity class.

Quick Tip

It's crucial to know the best, average, and worst-case time complexities for major sorting algorithms. - $O(n \log n)$ worst-case: Merge Sort, Heap Sort. - $O(n^2)$ worst-case: Bubble Sort, Insertion Sort, Selection Sort, Quick Sort. - Note that while Quick Sort's worst case is $O(n^2)$, its average case is $O(n \log n)$, which is why it's often used in practice.

---

5. Which of the following is true about the SQL JOIN operation?

(A) It combines rows from two or more tables based on a related column

(B) It only works on columns with unique values

(C) It is used for updating data in a table

(D) It can only be used to select data from one table at a time

Correct Answer: (A) It combines rows from two or more tables based on a related column

Solution:
Let's analyze each statement regarding the SQL 'JOIN' operation.

(B) This is incorrect. A 'JOIN' is frequently performed on foreign key columns, which often contain duplicate values.

(C) This is incorrect. The primary statement for updating data is 'UPDATE'. While a 'JOIN' can sometimes be used within an 'UPDATE' statement, its fundamental purpose is not updating data.

(D) This is incorrect. The purpose of a 'JOIN' is specifically to combine data from two or more tables. A simple 'SELECT' statement is used for a single table.

(A) This is correct. The 'JOIN' clause is used in a 'SELECT' statement to query data from multiple tables by creating a result set based on a logical relationship (a related column) between them.

Quick Tip

Think of 'JOIN' as the SQL tool for re-linking related data that has been separated into different tables through normalization. The most common type is 'INNER JOIN', which returns only the rows where the join condition is met in both tables.

6. Which of the following is NOT a characteristic of the C programming language?

(A) It supports low-level memory manipulation

(B) It is a high-level, structured language

(C) It allows direct access to hardware

(D) It does not support recursion

Correct Answer: (D) It does not support recursion

Solution:
Let's examine the characteristics of the C programming language.

(A) C supports low-level memory manipulation through the use of pointers, which is a major characteristic.

(B) C is a structured, high-level language, though it can also be used for low-level tasks. This is a characteristic.

(C) C allows for direct access to hardware, often using pointers and memory-mapped I/O, which is why it is used for operating systems and embedded programming. This is a characteristic.

(D) Recursion is a programming technique where a function calls itself. C fully supports recursion.

Therefore, the statement "It does not support recursion" is false and is NOT a characteristic of C.

Quick Tip

Remember that C is a procedural, structured language that supports all control flow mechanisms, including iteration and recursion. Its power lies in its ability to handle both high-level constructs and low-level memory management via pointers.

7. In the context of network protocols, what does the acronym "TCP" stand for?

(A) Transfer Control Protocol

(B) Transport Control Protocol

(C) Transmission Control Protocol

(D) Time Control Protocol

Correct Answer: (C) Transmission Control Protocol

Solution:
In the context of computer networking and the TCP/IP protocol suite, the acronym TCP stands for Transmission Control Protocol.

TCP is a core protocol of the Internet Protocol Suite.

It operates at the transport layer and provides reliable, ordered, and error-checked delivery of a stream of bytes between applications running on hosts communicating over an IP network.

It is connection-oriented, ensuring that data is delivered reliably to the destination.

Quick Tip

Remember the difference between TCP and UDP. TCP stands for Transmission Control Protocol and is connection-oriented (reliable). UDP stands for User Datagram Protocol and is connectionless (unreliable but faster).

---

8. Which of the following is the correct sequence of memory hierarchy from fastest to slowest?

(A) Registers > Cache > RAM > Disk

(B) Disk > RAM > Cache > Registers

(C) RAM > Registers > Disk > Cache

(D) Registers > Disk > RAM > Cache

Correct Answer: (A) Registers > Cache > RAM > Disk

Solution:
The memory hierarchy in a computer system is structured based on speed and cost.

The faster the memory, the closer it is to the Central Processing Unit (CPU).

The hierarchy from fastest (and smallest) to slowest (and largest) is:

1. Registers: Located inside the CPU, fastest access time.

2. Cache Memory (L1, L2, L3): Small, fast memory between registers and RAM.

3. Main Memory (RAM): Slower and larger than cache, also called primary memory.

4. Secondary Storage (Disk/SSD): Slowest and largest, non-volatile storage.

The correct sequence from fastest to slowest is Registers > Cache > RAM > Disk.

> Quick Tip
>
> The memory hierarchy is a trade-off: speed is inversely proportional to size and cost. Registers are the fastest and most expensive per bit, while Disk is the slowest and cheapest per bit.

---

9. Which of the following is a feature of the "Stack" data structure?

(A) Follows a First-In-First-Out (FIFO) order

(B) Allows elements to be inserted and removed at both ends

(C) Follows a Last-In-First-Out (LIFO) order

(D) Does not allow random access to elements

Correct Answer: (C) Follows a Last-In-First-Out (LIFO) order

Solution:
A Stack is a linear data structure that follows a specific order for its operations.

Insertion (Push) and removal (Pop) operations can only be performed at one end, which is called the 'top' of the stack.

(A) This is the characteristic of a Queue.

(B) This is the characteristic of a Deque (Double-ended queue).

(C) The element that was inserted last is the first one to be removed. This principle is called Last-In-First-Out ($LIFO$). This is the defining feature of a Stack.

(D) While technically true, the $LIFO$ principle is the most fundamental feature of the Stack data structure.

Quick Tip

Remember the two fundamental linear data structures: Stack for $LIFO$ (think of a stack of plates) and Queue for $FIFO$ (think of a line of people).

---

10. In object-oriented programming, which of the following is true about "Polymorphism"?

(A) It allows objects of different classes to be treated as objects of a common superclass

(B) It refers to the ability of a class to inherit properties from more than one class

(C) It refers to the ability to define a function with the same name but different signatures

(D) It allows data to be encapsulated in the class

Correct Answer: (C) It refers to the ability to define a function with the same name but different signatures

Solution:
Polymorphism literally means "many forms" and is one of the four pillars of OOP.

(A) This describes Subtype or Inclusion Polymorphism (Dynamic Polymorphism), which is a true aspect of the concept.

(B) This describes Multiple Inheritance, which is a mechanism, not polymorphism itself.

(D) This describes Encapsulation, another pillar of OOP.

(C) This precisely defines Method Overloading (or Function Overloading), which is a form of Compile-time Polymorphism (Ad-hoc Polymorphism).

Since both (A) and (C) are correct aspects of Polymorphism, and (C) provides a clear and common definition for one of its types, it is a true statement.

Quick Tip

Polymorphism can be classified as Compile-time (Overloading) or Run-time (Overriding). Overloading is defining multiple functions with the same name but different parameter lists. Overriding allows a subclass to provide a specific implementation for a method already provided by one of its superclasses.

---

11. Which of the following algorithms is based on the divide and conquer technique?

(A) Selection Sort

(B) Quick Sort

(C) Insertion Sort

(D) Bubble Sort

Correct Answer: (B) Quick Sort

Solution:
The Divide and Conquer paradigm involves three steps: Divide, Conquer, and Combine.

(A) Selection Sort, (C) Insertion Sort, and (D) Bubble Sort are all simple comparison-based sorting algorithms.

These three algorithms typically have a time complexity of $O(n^2)$ and do not follow the divide and conquer strategy.

(B) Quick Sort works by:
1. Divide: Pick an element as a pivot and partition the array around the pivot.
2. Conquer: Recursively sort the sub-arrays.
3. Combine: The array is already sorted, so this step is trivial (or non-existent).

Quick Sort is a classic example of a divide and conquer sorting algorithm, with an average time complexity of $O(n \log n)$.

Quick Tip

The main sorting algorithms based on the Divide and Conquer technique are Quick Sort and Merge Sort. Both have an average-case time complexity of $O(n \log n)$.

---

12. Which of the following sorting algorithms is stable?

(A) Quick Sort

(B) Merge Sort

(C) Heap Sort

(D) Selection Sort

Correct Answer: (B) Merge Sort

Solution:
A sorting algorithm is considered stable if it preserves the relative order of records with equal keys in the sorted output.

(A) Quick Sort is generally unstable because the partitioning process can change the relative order of equal elements.

(C) Heap Sort is generally unstable because of the way elements are swapped to maintain the heap property.

(D) Selection Sort is generally unstable because it involves swapping the smallest element with the element at the current position, which can disrupt the relative order of equal elements.

(B) Merge Sort is a stable sorting algorithm because the merging process can be implemented to ensure that elements with the same value from the first list are placed before elements from the second list, thus preserving their original order.

Quick Tip

Remember the stability of common sorts: Stable sorts include Merge Sort, Insertion Sort, and Bubble Sort. Unstable sorts include Quick Sort, Heap Sort, and Selection Sort.

---

13. What is the worst-case time complexity of the Binary Search algorithm?

(A) $O(1)$

(B) $O(n)$

(C) $O(\log n)$

(D) $O(n^2)$

Correct Answer: (C) $O(\log n)$

Solution:
Binary Search is an efficient algorithm for finding an item from a sorted list of n elements.

It works by repeatedly dividing the search interval in half.

In each comparison step, the problem size is reduced by half.

The worst-case scenario is when the element is either found in the last comparison or is not present at all.

The number of comparisons required in the worst case is proportional to $\log_2 n$.

Therefore, the worst-case time complexity of Binary Search is $O(\log n)$, which is highly efficient.

Quick Tip

Binary Search is an algorithm whose time complexity is consistently $O(\log n)$ in the best, average, and worst cases because the problem space is halved in every step, regardless of the target element's position.

---

14. Which of the following is a characteristic of the "Queue" data structure?

(A) Follows a Last-In-First-Out (LIFO) order

(B) Allows insertion at both ends

(C) Follows a First-In-First-Out (FIFO) order

(D) Does not allow deletion of elements

Correct Answer: (C) Follows a First-In-First-Out (FIFO) order

Solution:
A Queue is a linear data structure that operates like a real-world queue or line.

Insertion, called Enqueue, is performed at the rear end.

Deletion, called Dequeue, is performed at the front end.

(A) This is the characteristic of a Stack.

(B) This is the characteristic of a Deque (Double-ended queue).

(D) Deletion is a mandatory operation (Dequeue) for a Queue.

(C) The element that was inserted first is the first one to be removed. This principle is called First-In-First-Out ($FIFO$). This is the defining characteristic of a Queue.

Quick Tip

The $FIFO$ principle (First-In-First-Out) is what defines a Queue. Think of $FIFO$ for a Queue and $LIFO$ (Last-In-First-Out) for a Stack.
